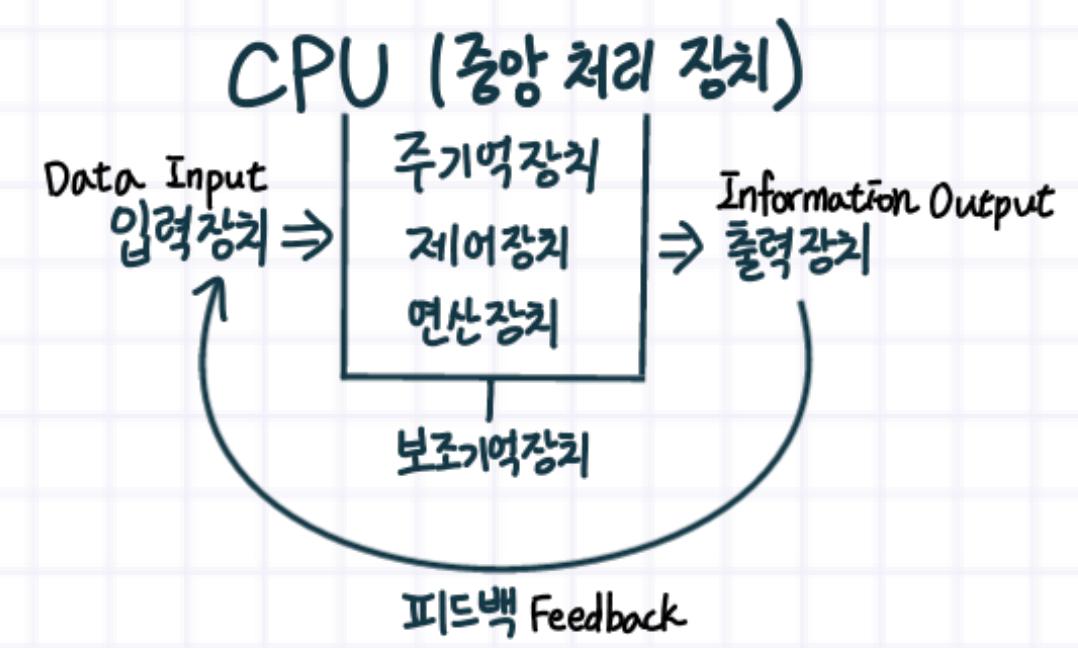


컴퓨터: 프로그램(명령어)이 지시하는 절차에 따라 자동적으로 대량의 데이터를 고속으로 처리하는 장치

컴퓨터의 특징

1. 신속성: 주어진 환경에서 아무 고장 없이 담당 기능 및 문제 처리를 원활하게 수행할 수 있는 척도.
 2. 호환성: 서로 다른 컴퓨터간에도 프로그램이나 자료의 공유가 가능함.
 3. 범용성: 일부에 국한되지 않고 다목적 분야에서 사용됨.
 4. 정확성: 컴퓨터에서 프로그램에 의해 처리된 결과는 정확함.
 5. 신속성: 컴퓨터에 의한 처리 속도는 매우 빠름.
 6. 자동성: 작성된 프로그램을 이용하여 자동으로 처리가 가능함.
 7. 대용량성: 멀티미디어 관련 자료 등 대량의 자료 처리 및 저장이 가능함.
- ~~창조성 가능성 안정성~~



자료 처리에 의한 분류

디지털 컴퓨터	셀 수 있는 이산 데이터(숫자, 문자 등)	논리 회로 사칙 연산	느림
아날로그 컴퓨터	셀 수 없는 연속적인 물리량(전류, 온도, 속도)	증폭 회로 미적분	연산 빠름

하드웨어 : 컴퓨터의 기계적인 부분. 본체, 모니터, 프린터, 키보드, 마우스 등을 통틀어 하드웨어라 칭함.

- 중앙 처리 장치: 제어 장치, 연산 장치, 주기억 장치
- 주변 장치: 입력 장치, 출력 장치, 보조 기억 장치

중앙 처리 장치(CPU: Central Processing Unit)

- 컴퓨터의 중추적인 역할, 각 부분의 동작을 제어하고 연산을 수행.
- 제어 장치와 연산 장치로 구성

레지스터 (Register)

- 1) 중앙 처리 장치 내의 고속 임시 기억 장치
- 2) 자료를 일시적으로 기억
- 3) 새로운 데이터가 전송되면 먼저 내용은 지워지고 새로운 내용만 기억
- 4) 사용 목적: 연산 속도의 향상
- 5) 워드 크기와 메모리의 용량에 따라 크기가 달라짐
- 6) 플립플롭(1비트 기억 소자)의 모임

제어 장치 (Control Unit)

- i. 입력, 출력, 연산, 기억 장치 등을 감시 감독
- ii. 프로그램의 명령을 해독하여 각 장치에게 처리하도록 지시
- iii. 제어 신호를 발행하여 명령어의 처리가 순서적으로 이루어지게 함

연산 장치 (Arithmetic & Logic Unit)

- 프로그램의 사칙, 논리 연산을 수행하고 비교 및 판단, 데이터의 이동, 편집 등을 수행

ACC (ACCUmulator)

누산기
산술 및 논리 연산의 결과를 일시적으로 기억

가산기 (Adder)

누산기와 데이터 레지스터의 값을 더하여 누산기에 저장

데이터 레지스터 (Data Register)

연산에 사용되는 데이터의 일시적인 저장을 위해 사용되는 레지스터

상태 레지스터 (Status Register) PSW(Program Status Word)

현재 상태를 나타내는 레지스터
각 비트별로 조건을 할당

보수기 (Complement)

뺄셈이나 나눗셈 연산을 위해 보수로 바꾸어 가산하는 장치

MAR (Memory Address Register)

기억 번지 레지스터
기억 장소의 주소를 기억하는 레지스터

MBR (Memory Buffer Register)

기억 버퍼 레지스터
기억 장치를 통해 접근되는 정보의 내용을 기억하는 레지스터

IR (Instruction Register)

명령 레지스터
현재 수행중인 명령어를 기억하는 레지스터

PC (Program Counter)

프로그램 카운터
다음에 수행할 명령어의 번지를 기억하는 레지스터

명령 해독기 (Instruction Decoder)

IR에 기억된 명령들을 해독해서 각 장치에 제어 신호를 보냄

부호기 (Encoder)

중앙 처리 장치에서 실행하기 위한 전기 신호로 변환하여 각 장치에 보내는 기능

소프트웨어(S/W) : 하드웨어를 움직여주는 프로그램. 시스템 소프트웨어와 응용 소프트웨어로 구성.

시스템 소프트웨어: 컴퓨터 시스템의 전반적 운영을 위한 기본적인 소프트웨어.

- 운영체제, 언어 번역기, 유틸리티 프로그램

운영체제(OS: Operating System)

- 컴퓨터 하드웨어의 성능을 최대한 효율적으로 운영하기 위해 하드웨어와 사용자 사이에 있는 프로그램
 - 제어 프로그램과 처리 프로그램으로 구성

제어 프로그램 (Control Program)

1. 감시 프로그램 (Supervisor Program)	컴퓨터 시스템 전체의 작동 상태를 감시, 감독하는 프로그램
2. 작업 관리 프로그램 (Job Management Program)	작업 관련 데이터의 준비와 처리를 관리하는 프로그램
3. 데이터 관리 프로그램 (Data Management Program)	여러 종류의 데이터와 파일을 관리해주는 프로그램

처리 프로그램 (Process Program)

1. 언어 번역 프로그램 (Language Translator Program)	기계어로 번역하기 위한 프로그램
2. 서비스 프로그램 (Service Program)	유ти리티, 정렬/병합 프로그램과 같이 사용 빈도가 높은 프로그램들을 제작회사에서 미리 프로그램화하여 제공하는 프로그램
3. 문제 처리 프로그램 (Problem Processing Program)	사용자가 업무에 적용하여 그에 따라 작성한 프로그램

언어 번역기

1. 컴파일러(Compiler): 고급 언어를 기계어로 번역하는 프로그램으로 전체를 한 번에 번역.
 - 전체를 한번에 번역, 목적 프로그램 생성, 수행 속도 빠름(효율성)
 2. 어셈블러(Assembler): 어셈블리(Assembly) 언어를 기계어로 번역하는 프로그램.
 3. 인터프리터(Interpreter): 대화식 언어로 작성된 프로그램을 필요할 때마다 매번 기계어로 번역하여 실행하는 프로그램.
 - 행 단위로 번역, 목적 프로그램 미생성, 수행 속도 느림

언어 번역 과정

원시 프로그램 -----> 목적 프로그램 -----> 로드 모듈 -----> 실행

번역

연계 편집 적재(Loder)

적재(Loder)

- 1) 원시 프로그램 : 사용자가 프로그램 언어(고급 언어, 어셈블리어)로 작성한 프로그램
 - 1-1) 언어 번역기 : 특정 프로그래밍 언어로 작성된 내용을 컴퓨터가 이해할 수 있는 기계어로 바꾸어주는 프로그램
 - 2) 목적 프로그램 : 컴파일러에 의해 기계어로 번역된 프로그램
 - 2-1) 연계 편집 : 목적 프로그램을 실행 가능한 프로그램으로 만드는 과정
 - 3) 로드 모듈 : 실행 가능한 상태의 프로그램
 - 3-1) 로더 : 로드 모듈 프로그램을 주기억 장치 내로 옮겨서 실행해주는 소프트웨어.
 - 할당(Allocation), 연결(Linking), 재배치(Relocation), 적재>Loading

응용 소프트웨어 : 실제 업무처리를 위해 개발된 프로그램을 의미. ex) 워드프로세서, 스프레드시트, 프레젠테이션, 데이터베이스 등

펌웨어 : ROM에 소프트웨어를 접목시켜 저장한 중간 형태.

- 하드웨어를 교체하지 않아도 소프트웨어를 업그레이트하여 시스템 성능을 향상시킬

불 대수

- 2진수의 값으로 논리적 동작을 취급하는 대수
- 하나의 변수는 0 또는 1의 값.
- 연산자: 논리곱(AND), 논리합(OR), 논리 부정(NOT)

기본 성질

합의 법칙

$$\begin{cases} X + 0 = X \\ X + 1 = 1 \\ X + X = X \\ X + \bar{X} = 1 \end{cases}$$

곱의 법칙

$$\begin{cases} X \cdot 0 = 0 \\ X \cdot 1 = X \\ X \cdot X = X \\ X \cdot \bar{X} = 0 \end{cases}$$

교환법칙

$$\begin{cases} X + Y = Y + X \\ X \cdot Y = Y \cdot X \end{cases}$$

흡수법칙

$$\begin{cases} X + X \cdot Y = X \\ X + \bar{X} \cdot Y = X + Y \\ X \cdot (X + Y) = X \end{cases}$$

결합법칙

$$\begin{cases} X + (Y + Z) = (X + Y) + Z \\ X \cdot (Y \cdot Z) = (X \cdot Y) \cdot Z \end{cases}$$

분배법칙

$$\begin{cases} X \cdot (Y + Z) = (X \cdot Y) + (X \cdot Z) \\ X + (Y \cdot Z) = (X + Y) \cdot (X + Z) \end{cases}$$

대합성의 법칙: $\bar{\bar{X}} = X$

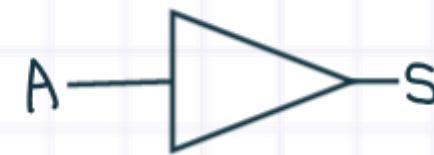
드모르간의 법칙

$$\begin{cases} \overline{X+Y} = \overline{X} \cdot \overline{Y} \\ \overline{X \cdot Y} = \overline{X} + \overline{Y} \end{cases}$$

기본 논리 회로

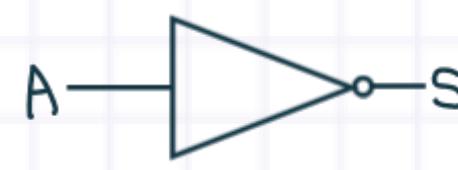
0. Buffer 게이트

입력값 그대로 출력



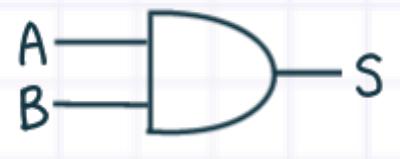
0-1. NOT 게이트 = Inverter 게이트

입력값의 반대값이 출력



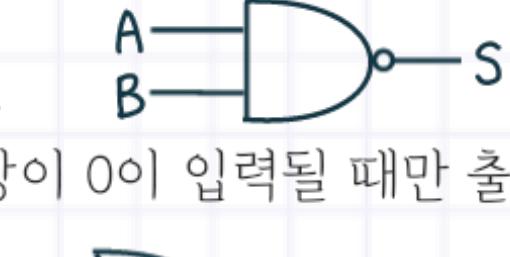
1. AND 게이트

두 개의 입력값이 모두 1일 때만 출력값 1



1-1. NAND 게이트

두 수 중 하나 이상이 0이 입력될 때만 출력값 1 (AND 결과의 부정)



2. OR 게이트

두 개의 입력값 중 하나 이상이 1이면 출력값 1



2-1. NOR 게이트

두 수 모두 0이 입력될 때만 출력값 1 (OR 결과의 부정)



3. XOR 게이트 = eXclusive OR

둘 중 하나의 값이 1일 때만(서로 다를 때) 출력값 1



3-1. XNOR 게이트 = eXclusive NOR

두 수 모두 0 또는 1일 때만(같을 때) 출력값 1



조합 논리 회로: 반가산기, 전가산기, 감산기, 인코더, 디코더, 멀티플렉서 등

반가산기(HA): AND 게이트와 XOR 게이트로 구성

전가산기(FA): 2개의 반가산기와 1개의 OR 게이트로 구성

멀티플렉서(Multiplexer): 2^n 개의 입력을 받아들여 하나의 출력선으로 정보를 출력하는 논리 회로

순서 논리 회로: 조합 논리 회로와 1비트 기억 소자인 플립플롭으로 구성. 기억 능력 0

- RS 플립플롭, JK 플립플롭, D 플립플롭, T 플립플롭

RS 플립플롭		JK 플립플롭		D 플립플롭		T 플립플롭	
S	R	J	K	D	Q _{t+1}	T	Q _{t+1}
0	0	0	0	0	Q _t	0	Q _t
0	1	0	1	1	Q _t	1	Q _t
1	0	1	0	0	Q _t	0	Q _t
1	1	1	1	1	Q _t	1	Q _t

※ RS, JK, D, T 플립플롭은 초기 상태를 설정하는 경우에만 사용된다.

※ JK 플립플롭은 초기 상태를 설정하는 경우에만 사용된다.

※ D 플립플롭은 초기 상태를 설정하는 경우에만 사용된다.

※ T 플립플롭은 초기 상태를 설정하는 경우에만 사용된다.

진수 표현

- 2진수(1자리 1비트) : 0과 1로 구성된 수
- 8진수(3비트) : 0~7 사이의 숫자로 구성된 수
- 10진수 : ~9 사이의 숫자로 구성된 수
- 16진수 : 0~15 사이의 숫자로 구성된 수 [10~15는 A~F]

① 10진수 → 다른 진수로 변환

$$\begin{array}{c}
 (17)_{10} \\
 \begin{array}{r}
 \overline{2} \overline{1} \overline{1} \\
 \overline{2} \overline{1} \overline{0} \cdots 1 \\
 \overline{2} \overline{1} \overline{0} \cdots 0 \\
 \hline
 2\text{진수: } (10001)_2
 \end{array}
 \quad
 \begin{array}{r}
 \overline{8} \overline{1} \overline{1} \\
 \overline{2} \cdots 1 \\
 \overline{8}\text{진수: } (21)_8
 \end{array}
 \quad
 \begin{array}{r}
 \overline{16} \overline{1} \overline{1} \\
 \overline{1} \cdots 1 \\
 \overline{16}\text{진수: } (11)_{16}
 \end{array}
 \quad
 \begin{array}{r}
 \overline{(10.375)}_{10} \\
 \begin{array}{r}
 \overline{2} \overline{1} \overline{0} \cdots 0 \\
 \overline{2} \overline{1} \overline{0} \cdots 1 \\
 \overline{1} \cdots 0 \\
 \hline
 2\text{진수: } (1010.011)_2
 \end{array}
 \quad
 \begin{array}{r}
 \overline{0.375} \\
 \overline{0.750} \\
 \overline{1.5} \\
 \hline
 1.0
 \end{array}
 \end{array}
 \end{array}$$

$$\begin{array}{l}
 (10001)_2 = 1 \times 2^4 + 1 \times 2^0 = (17)_{10} \\
 (21)_8 = 2 \times 8^1 + 1 \times 8^0 = (17)_{10} \\
 (11)_{16} = 1 \times 16^1 + 1 \times 16^0 = (17)_{10} \\
 (12.5)_8 = 1 \times 8^1 + 2 \times 8^0 + 5 \times 8^{-1} = 10 + \frac{5}{8} = 10.625
 \end{array}$$

③ 2진수 → 8진수

16진수 → 2진수

2 5 6 . A C

0010 1011 0101 1100

2진수 → 16진수

1 1 3 . 6 6

0010 0011 1101 0110

+ ④ 2진수 → 그레이 코드

$\begin{array}{r} 1011 \\ \downarrow \downarrow \downarrow \\ 1110 \end{array}$

그레이 코드 → 2진수

$\begin{array}{r} 1110 \\ \uparrow \downarrow \downarrow \\ 1011 \end{array}$

명령어의 구성 : 명령 코드부(OP-Code)와 주소부(Operand, 번지부)

[명령어 형식]

0-주소 형식 = 스택 구조

- 주소부(오퍼랜드부) 없이 데이터가 명령어 자체에 있는 방식
- 스택 구조의 컴퓨터에서 사용
- 연산 속도가 가장 빠름

1-주소 형식 = ACC(누산기) 구조

- 주소부가 한 개.
- 데이터의 처리를 위해 누산기 구조의 컴퓨터에서 사용

2-주소 형식 = 범용 레지스터 구조

- 주소부가 두 개.
- 연산 후 데이터 값 보존 X.

3-주소 형식 = 범용 레지스터 구조

- 주소부가 세 개.
- 원래 값 보존.

주소 지정 방식

명령어의 길이(짧→길): 즉시주소→직접주소→간접주소

메모리 참조 횟수(적→많): 즉시주소→직접주소→간접주소

처리 속도(느→빠): 간접주소→직접주소→즉시주소

1. 묵시적 주소 지정(Implied Addressing)

- 스택 구조의 0-주소 방식.

2. 즉시 주소 지정(Immediate Addressing)

- 명령어 주소 부분에 있는 값 자체가 실제 데이터가 됨.

3. 직접 주소 지정(Direct Addressing)

- 주소 부분에 있는 값이 실제 데이터가 있는 주기억 장치 내 주소를 나타냄.

4. 간접 주소 지정(Indirect Addressing)

- 주소 부분으로 지정한 기억 장소의 내용이 실제 데이터가 있는 곳의 주소로 사용됨.

제어 장치의 제어 방식

인출 사이클(Fetch Cycle): 주기억 장치로부터 CPU가 무엇을 하고 있는지 나타냄.

간접 사이클(Indirect Cycle): 간접 주소 지정이 허용되면 유효 번지를 읽기 위해 기억 장치를 한 번 더 접근함.

실행 사이클(Execute Cytle): 인출된 명령어를 이용하여 직접 명령을 실행함.

인터럽트 사이클(Interrupt Cycle): 인터럽트가 발생했을 때 처리함.

주기억 장치(Main Memory Unit)

1. ROM(Read Only Memory): 읽기만 가능한 기억 장치, 비휘발성 기억.

- 1) Mask ROM - 제조 회사에서 제작할 때 기억 장치 내용 변경이 불가능.
- 2) PROM(Programmable ROM) - 한 번에 한해서 사용자가 직접 원하는 정보를 기록할 수 있으나 변경 불가능.
- 3) EPPROM(Erasable PROM) - 자외선을 이용해 삭제 & 재기록 가능
- 4) EEPROM(Electrically EPROM) - 전기적인 방법으로 삭제 & 재기록 가능

2. RAM(Random Access Memory): 읽기 쓰기가 가능한 기억 장치, 휘발성 메모리.

- 1) SRAM(Static RAM) - 소비 전력 많고 속도 빠름.
 - 캐시 메모리(Cache Memory)로 사용.
- 2) DRAM(Dynamic RAM) - 재충전(Refresh) 필요.
 - SRAM보다 속도 느림.

기억 장치 관리 - 페이지 교체 기법

FIFO(First In First Out): 가장 먼저 들어온, 가장 오래된 페이지를 교체할 페이지로 선택.

LRU(Least Recently Used): 가장 오랫동안 사용되지 않은 페이지를 선택.

LFU(Least Frequently Used): 사용된 횟수가 가장 적은 페이지를 선택.

NUR(Not Used Recently): 사용되지 않은 페이지를 선택.

최적화 기법(OPT: OPTimal replacement): 사용되지 않거나 사용도가 낮은 페이지를 선택.

데이터베이스 디자인 단계 순서

1. 목적 정의 ->
2. 필요한 테이블을 정의 ->
3. 테이블에서 필요한 필드를 정의 ->
4. 테이블 간 관계 정의

데이터베이스 설계 단계

요구 분석 -> 개념적 -> 논리적 -> 물리적 설계 -> 구현

windows 바로가기키

F1: 도움말 실행, F2: 이름 바꾸기, F3: 찾기(파일, 폴더)

F4: 주소 표시줄, F5: 새로고침,

F6: 탐색기 왼쪽 오른쪽 구역 이동

F8: window 부팅시 부팅 메뉴 나타남

Ctrl + C: 복사하기, Ctrl + V: 붙여넣기, Ctrl + X: 잘라내기

Ctrl + A: 모두 선택, Ctrl + Z: 방금 전 실행 취소

Ctrl + Esc: 시작 메뉴 호출

Alt + F4: 실행중인 현재 창 종료

Alt + Tab: 실행 중 프로그램 간 작업 전환

Shift + F10: 프로그램이나 선택 항목 바로가기 메뉴

Shift + Delete: 휴지통 안 거치고 바로 삭제

Print Screen: 화면 전체 복사

Alt + Print Screen: 현재 사용중 활성창 복사

Ctrl + Alt + Delete: 프로그램이나 시스템 강제 종료

Shift + CD삽입: CD-ROM의 자동 실행 방지

UNIX

커널(Kernel): UNIX의 가장 핵심적인 부분

셸(Shell): 사용자와 UNIX 간의 인터페이스
= DOS의 COMMAND.COM

c shell = %

korn shell = \$

boume shell = #

i-node: 파일 정보 규정 자료 구조.

시스템 및 파일 관련 명령어

who: 현재 시스템 사용자 표시

date: 현재 날짜 표시

cal: 연도별 달력 표시

kill: 수행중 프로세스 강제 중지

ps: 프로세스 상태 보기

sleep: 프로세스 수행 일시 중지

shutdown: 작업중 상태 체계적 종료

open: 지정된 형식으로 파일 열기

close: 파일 닫기

read: 파일 내용 순차적 열기

write: 파일에 내용을 순차적 기록

finger: 등록 사용자 정보 조회

chown: 파일 소유권 변경

tar: 여러 개 파일 하나로 묶음

wc: 줄 단어 수, 바이트 수 세기

more: 파일 리스트 한번에 한페이지

cat: 간략하게 출력

history: 지금까지 입력 명령어 표시

cmp: 두 파일 파일점 바이트, 행 번호 표시

DOS(Disk Operating System) : 개인용 컴퓨터에서 디스크 관리 및 파일 관리 등을 다루기 위한 운영체제
- CUI(Character User Interface) 방식

부팅 시 필요한 파일: IO.SYS, MSDOS.SYS, CONFIG.SYS, COMMAND.COM, AUTOEXEC.BAT

부팅 종료: 콜드 부팅(하드웨어적 부팅, 리셋버튼으로 재부팅), 웜부팅(소프트웨어적 부팅, Ctrl+Alt+Delete로 재부팅)

DOS 명령어 = [unix 명령어]

내부 명령어

1. dir(파일 목록 보기) = [ls]
2. del(파일 삭제) = [rm]
3. ren, rename(파일 이름 변경) = [mv]
4. type(텍스트 파일 내용 보여줌) = [cat]
5. prompt(프롬프트 설정)
6. md(mkdir, 디렉터리 생성) = [mkdir]
7. cd(chdir, 경로 변경) = [cd]
8. rd(rmdir, 디렉터리 삭제) = [rmdir]
9. path(경로 설정 및 해제)
10. ver(dos의 버전 표시)
11. vol(드라이브의 볼륨명, 일련번호 표시)
12. cls(화면 내용 지움)
13. date(날짜 확인 및 설정)
14. time(시간 확인 및 설정)
15. copy(파일 복사) = cp
16. exit(이전 프로세스로 복귀)

외부 명령어

1. format(디스크 초기화)
2. fdisk(논리적인 파티션 설정)
3. sys(부팅 디스크 생성)
4. chkdsk(디스크 상태 점검)
5. attrib(파일 속성 변경) = [chmod]
6. deltree(파일 및 하위 디렉터리까지 삭제)
7. move(파일, 디렉터리 변경 및 이동)
8. unformat(포맷한 디스크 복구)
9. backup(파일 손상에 대비해 데이터 복사)
10. diskcopy(디스크 복사)
11. xcopy(파일, 디렉터리 및 하위 디렉터리 복사)
12. undelete(삭제했던 파일 복구)
13. find(특정 문자열 검색)
14. sort(정렬해 결과를 화면, 파일 형태로 출력)
15. more(화면 단위 출력) = [more]
16. label(볼륨명 지정)
17. restore(백업 데이터 복구)
18. fc(두 개의 파일을 비교해 차이 표시)

정보 전송 부호

Baudot 코드 : 5비트(32), 텔레스통신에서 이용

BCD 코드: 6비트 코드(64), Zone Bit 2비트, Digit Bit 4비트, 영소문자 구별X

ASCII 코드: 7비트 코드(128), Zone Bit 3비트, Digit Bit 4비트, 영소문자 구별O, 데이터통신용으로 사용

EBCDIC 코드: 8비트 코드(256), Zone Bit 4비트, Digit Bit 4비트, 범용 컴퓨터에서 사용

디지털 데이터의 아날로그 부호화

- 1) 진폭 편이 변조(ASK, Amplitude Shift Keying): 서로 다른 진폭을 적용해 변조
- 2) 주파수 편이 변조(FSK, Frequency Shift Keying): 서로 다른 주파수를 적용해 변조
- 3) 위상 편이 변조(PSK, Phase Shift Keying): 서로 다른 위상을 적용해 전송
- 4) 진폭 위상 변조(QAM, Quadrature Amplitude Modulation): 진폭과 위상을 변조해 전송하는 방식

OSI 참조 모델 7계층

하위층	상위층
1계층, 물리 계층	매체 접근에 따른 기계적, 전기적, 물리적 절차를 규정
2계층, 데이터 링크 계층	인접 개방형 시스템 간의 정보 전송 및 오류 제어
3계층, 네트워크 계층	정보 교환, 중계 기능, 경로 선정, 유통 제어 등
4계층, 전송 계층	송수신 시스템 간의 논리적 안정 및 균등한 서비스 제공
	응용 프로세스 간의 연결 접속 및 동기 제어 기능
	정보의 형식 설정 및 부호 교환, 암호화, 해독, 압축 등
	응용 프로세스 간의 정보 교환 및 전자 사서함, 파일 전송 등